Introduction to Stata

Katrien Stevens
k.stevens@ucl.ac.uk

I Getting Started

- what is Stata?

Stata is a fast and user friendly statistical package, which provides comprehensive data management and analysis capabilities. **Stata** offers a wide array of pre-defined statistical procedures, yet its programming features allow for much flexibility.

Stata reference manuals are available in the library. The help function within the program is very useful and almost equivalent to the information in the manuals.

- **Stata** for Windows:

    xstata /* to use in x-windows environment */

Stata for Windows uses pull-down menus, which are easy to use.

- exiting Stata:

    exit, clear /* clear is necessary if a dataset is currently loaded */

- using on-line help:

    help subject /* search and lookup perform similar functions */

- using on-line tutorials:

    tutorial nameoftutorial /* tutorials include intro, contents, graphics, tables, regress, anova, logit, survival, factor, ourdata, yourdata */

- stopping execution:

    q   OR   press break-key

- abbreviations:

You can abbreviate commands in **Stata**. However, there is no rule for abbreviations. Some commands are uniquely identified with only one letter, some require a full name and will not accept abbreviation

II Basics

- command syntax

In general, **Stata** commands will have the following format (terms in brackets are not always required):

command [varlist] [if exp] [in range] [weight] [, options]

- varlist: specifies variables to be used by a given command, if blank, all variables are used;
- if exp: chooses only observations which satisfy a given condition exp;
- in range: specifies a range of observations to be used;
- weight: indicates the type of a weighting scheme to be used;
- options: command-specific options

example 1:

```
cd "c:\program files\stata\"
pwd
use auto
summarize gratio trunk
sort foreign
by foreign: summarize gratio trunk
```

example 2:

```
summarize price length if foreign==0
summarize price length if price>10000
summarize price length in 1/25, detail
```

- keeping logs

Stata can save each session into a log file. Contents of the log file can be printed or easily copied and pasted into other applications (e.g. Microsoft Word, Microsoft PowerPoint). **Stata** for Windows has a pop-up menu that makes log file management very straightforward. However, the following will also work:

```
log using filename /* to open a log file, options: append, replace, noproc */
log off  /* to temporarily suspend a log file */
log on  /* to resume writing to a log file */
log close  /* to close a log file */
type filename.log  /* to view contents of a log file */
```

- loading datasets into Stata

Loading datasets into **Stata** can be a very frustrating task. Following a few simple rules will make this task easier. The best dataset format to use in **Stata** is .dta. You can use StatTransfer to create .dta files. If this resource is not available, you can read your data directly into **Stata**:

```
insheet using filename   /* original file with tab and comma delimiters, no space delimiters*/
```

E.g. an excel-file with the data can be saved in '.csv'-format (comma delimited) + use insheet to import in Stata.

Whenever the insheet command cannot be used, you will have to use either of two commands: infile or infix.

This will also involve using a data dictionary, which is beyond the scope of this class. For more information on infile and infix, please refer to **Stata** manuals.

input /* for manual input */

```
    input x y
    1.  2   3
    2.  9   8
    3.  end
```

- Saving datafiles

save filename , replace /* saves file as filename, replace needed if a file of that name already exists: overwrites an existing datafile */

- do-files

A do-file is an ASCII text file, which is executed when you type:

do filename

In **Stata** for Windows, use the do-file editor to create a do-file. Typically, do-files store sequences of **Stata** commands. For example, if your file (myprogram.do) contains:

```
    use auto
    sum price
    describe mpg weight
```

you will type:

do myprogram.do /* to execute */

- ado-files

Ado-files define **Stata** commands, but some commands are built-in, rather than defined by an ado-file.

Ado-files containing new procedures can be obtained from the **Stata** web site and other users, and easily added into the appropriate **Stata** directory on your computer. Some useful commands to deal with ado-files:

sysdir /* to get a listing of **Stata** directories */
which logistic /* to find the location of logistic.ado */
type logistic.ado /* to view the code */

- setting the size of memory

By default, **Stata** allocates 1 megabyte to data areas. To change it, use:

set memory 20000 /* this gives you 20K*/

- Controlling output

-more- may appear in your results window when you try to output a long listing

To see the next line: press Enter
To see the next screen: press any key

Set more off / on /* to switch the more-command off/on */

III Data Management

- describing datasets

You can easily describe data in **Stata**. Some useful commands include:

label /*to change a description of a variable */
label variable price "Price in U.S. dollars"

describe /* to describe a format of a variable */
de price

list /* to list observations or variables */
list price trunk

count /* to obtain a count for a given condition */
count if price < 5000

summarize /* produces summary statistics – detailed*/
su year
su year, det

tabulate /* produces one and two-way frequency counts */
tab year
tab year gender

table /* produces a table of summary statistics */
table price

Note: *by*-command: the command is repeated for every value of the variable specified (make sure the variable is sorted)
sort region
by region: su price

- data manipulation

In **Stata** for Windows, you can manipulate data directly in the data editor. Some common-task commands include:

generate / * to create a new variable */
*generate newprice=price*1.2*

Note: *egen* (egenerate): extensions to generate – to create means, standard deviations, sums,… of existing variables

replace /* to replace an existing variable */
replace newprice=. if newprice < 10000

rename /* to rename an existing variable */
rename newprice nprice

drop /* to delete a variable */
drop turn

keep /* works in the opposite way to drop */
keep in 2/l /* deletes the first observation */

sort /* to sort variables in ascending order – note: *gsort*: ascending or descending order*/
sort price
gsort + year - price

- logical operators: & (and), | (or), ~ (not)

  list if price>13500 | (price<4500 & mpg<19)

- matrix operations

Stata provides a variety of matrix functions. Some examples follow:

matrix A=(1,3,4\9,8,0) /* to create a matrix */
matrix list A /* to view a matrix */
*matrix B=A'*A* /* to compute inner product */
matrix colnames A = price weight length /* to label columns */
matrix rownames A = Ford Chrysler /* to label rows */
matrix define C=I(10) /* to create an identity matrix */
display trace(C) /* to calculate the trace of a matrix */
display det(C) /* to calculate the determinant of a matrix */
matrix D=J(10,8,5) /* to create a matrix of constants */

IV Statistical Procedures

Stata offers a multitude of statistical procedures. Some important issues include:

- accessing previous results

Stata saves results and makes them easily accessible. Last estimation results are stored in an array called e( ). Other results are saved in r(name).

example 1:

```
reg mpg weight
estimates list
display "The F statistic is " e(F)
```

example 2:

```
sum price
return list
gen pricedev=price-r(mean)
```

- generating lags and leads

For time-series analysis, you can create lags and leads of a variable.

```
gen prlag1 = price[_n-1]
```

- ordinary least squares

It is very easy to run a regression in **Stata**.

```
regress price mpg if foreign in 1/70 /* only first 70 obs used */
regress mpg price, level (67) /* 67% confidence level */
vce /* variance-covariance matrix */
mat list e(V) /* same as vce */
predict pricehat /* to create predictions based on estimates */
gen newvar=_b[_cons]+_b[mpg]*mpg /* to access reg estimates */
```

- hypothesis testing

The syntax for hypothesis testing is also straightforward.

example 1 (t-test, F-test):

```
regress price mpg weight
test mpg weight
test mpg = 2, accum
test 2*mpg = mpg – weight
```

example 2 (likelihood-ratio test):

regress price mpg length weight gratio
lrtest, saving(0)
regress price mpg
lrtest

example 3 (non-linear Wald test):

regress price mpg length
testnl _b[mpg]^2 + _b[length]=0

- other estimation commands (a narrow selection)

probit
logit
tobit
ivreg /* instrumental variables */
nl /*non-linear ordinary least squares */

V Graphs

- graph command syntax

command [varlist] [=exp] [if exp] [in range] [weight] [, options]

examples:

graph7 price length
graph7 mpg weight if foreign
graph7 price mpg in 1/50

Note: The '7' refers to a command of an earlier version of stata (stata7). In later versions of stata (8 & above), the graphics commands have become more complicated (and flexible):

graph twoway scatter price length
graph twoway line price length

- graph types

Stata supports the following graph types: histogram, twoway, matrix, oneway, box, star, bar, dot and pie.

graph7 mpg, bin(7) ylabel xlabel normal /* a histogram */
histogram mpg, bin(7) ylabel xlabel normal /* a histogram in later Stata-versions*/

- adding titles

```
graph7 mpg price,
        t1title(This is the t1title)
        t2title(This is the t2title)
        b1title(This is the b1title)
        b2title(This is the b2title)
        r1title(This is the r1title)
        r2title(This is the r2title)
        l1title(This is the l1title)
        l2title(This is the l2title)
```

- labeling axes

```
graph7 mpg displ, xline yline /* to create grid lines */
graph7 mpg displ, xlabel(75,275,475) xline(175,375)
graph7 mpg displ, noaxis
```

- specifying plotting symbols

You can specify a type of symbol to be used in **Stata** graphs. The list includes:

O (large circle)
S (large square)
T (large triangle)
o (small circle)
d (small diamond)
p (small plus)
. (dot)
i (invisible)
[varname] (contents of variable)
[_n] (observation numbers)

for example:

```
regress price mpg
predict pricehat
gr7 pricehat mpg, symbol(S)
graph7 price mpg, symbol([make])
graph7 mpg displ if foreign, s([make]) psize(125)
```

- connecting points

. (no connections, default)
l (straight lines between points)
L (straight lines between ascending x points)
m (median bands using straight lines)
s (median bands using cubic splines)
J (connect rectilinearly, making steps)

|| (connect two variables vertically)
II (same as ||, but cap bottom and top of line)

for example:

gr price mpg, connect(l) sort
gr price mpg, c(s) bands(8)
gr price pricehat mpg, symbol(Oi) connect(.l) sort

- multiple graphs

Stata allows to generate multiple graphs in one step.

graph7 mpg displ, by(foreign) total
graph7 mpg displ weight gratio, matrix label
graph7 price, bin(4) normal title(Prices) saving(part1)
graph7 length weight, title(Length vs. Weight) saving(part2)
graph7 using part1 part2, title(Both Graphs) saving(part3)

VI **Stata** Resources

There are several excellent **Stata** resources available on the Internet. They include program databases, discussion forum, and task-specific user web sites.

- Stata Corporation (www.stata.com)

This is a place to start your search. It offers an extensive selection of procedures, as well as links to other **Stata**-related sites.

- Survival Analysis with Stata

(http://www.iser.essex.ac.uk/teaching/degree/stephenj/ec968/) This is a site developed by Stephen Jenkins, who is a frequent contributor to the Stata discussion list. This site is devoted mainly to survival analysis.

- Stata Teaching and Learning Tools (www.ats.ucla.edu/stat/stata)

Hosted by UCLA, this site provides a number of programs which illustrate simple (and not so simple) statistical concepts.

- Stata Listserv (www.stata.com/support/statalist)

This is where you can exchange ideas and ask questions. It is probably the most effective resource for seeking help and guidance.